© Strojniški vestnik 46(2000)5,309-317
ISSN 0039-2480
UDK 007.52.001.41
Strokovni članek (1.04)
© Journal of Mechanical Engineering 46(2000)5,309-317
ISSN 0039-2480
UDC 007.52.001.41
Speciality paper (1.04)

Šolski robot SLR 1500 - razvoj in simulirni program

The SLR 1500 Training Robot - Development and Simulation Program

Juraj Uríček - Viera Poppeová - Róbert Zahoranský

Šolski robot SLR 1500 je bil razvit na Univerzi v Žilini in na IQM Zvolen (Slovaška) za pouk robotike na visokih šolah in univerzah. Njegov razvoj povezuje programiranje, tehnično kibernetiko, reševanje problemov vodenja, razvoj krmilnih sistemov za industrijske robote in delovanje numerično krmiljenih strojev. Šolski robot SLR 1500 ima rotacijske členke. Ima pet prostostnih stopenj, tri od njih so namenjene za zagotavljanje lege orodja v prostoru, preostali dve določata njegovo usmeritev glede na predmet, ki ga premika.

(Ključne besede: roboti učeči, razvoj robotov, programi simulacijski, kinematika robotov)

The SLR 1500 training robot was developed at the University of Žilina and at IQM Zvolen (Slovakia) for the teaching of robotics at high schools and universities. Its design incorporates assistance with the programming, the technical cybernetics, solving the drive control problems, the design of the control system for industrial robots and operation of NC machines. The SLR 1500 training robot has an angular design. It has five degrees of freedom and three of them serve to position the end effector in space and the remaining two determine its orientation with respect to the object being manipulated.

(Keywords: training robots, development of robots, simulation programs, robot kinemtics)

0 UVOD

Kinematična struktura šolskega robota SLR 1500 je sestavljena iz rotacijskega telesa, dveh ročic in zapestnega mehanizma. Vsi deli so povezani z rotacijskimi členki, tako da ročici nista v isti ravnini, ampak se premikata ena glede na drugo. Takšna konstrukcija dovoljuje izjemno veliko območje vrtenja posamezne ročice (sl. 1).

Parametri robota:

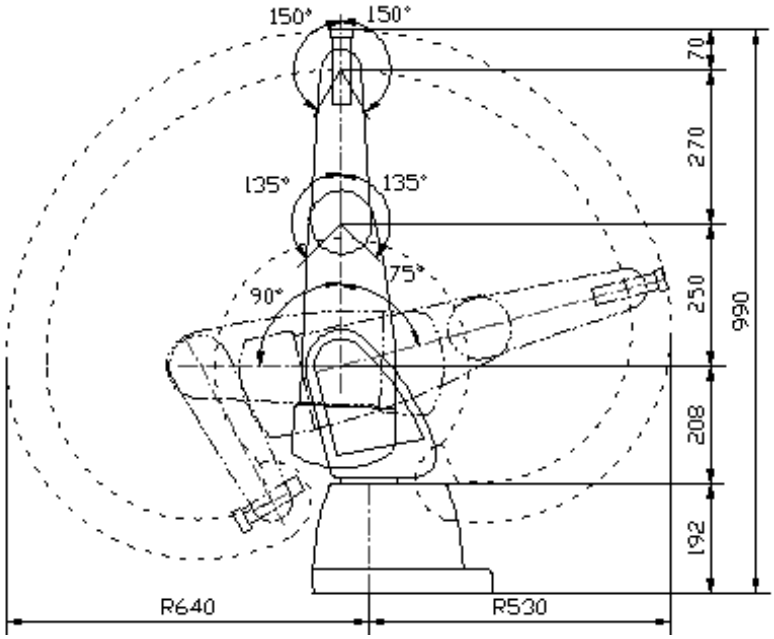
- število prostostnih stopenj: 5
- ponovljiva natančnost lege : 0,25 mm za ponovni gib ročice v eni smeri
- največji vodoravni doseg od rotacijske osi enote: 640 mm
- največji rotacijski kot vrtilne enote: 330°
- največji rotacijski kot spodnje ročice: 165°
- največji rotacijski kot zgornje ročice: 270°
- največji rotacijski kot zapestja: 300°
- največji prečni rotacijski kot zapestja: 270°
- največja delovna hitrost vrtilne enote, spodnje ročice: 130°/s
- zgornje ročice: 130°/s
- prečna rotacija zapestja: 130°/s
- rotacije zapestja: 300°/s
- pozicioniranje: koračni števci

0 INTRODUCTION

The kinematic structure of the SLR 1500 robot consists of a rotational waist, two arms and a wrist mechanism. All parts are connected with rotational joints in such a way that the arms are not in the same plane but are moved with respect to each other. Such a design allows an extremely large rotational range of the individual arms (Fig. 1).

The robot parameters:

- number of degrees of freedom: 5
- repeatable positional accuracy: 0.25 mm for repeated arm movement in one direction
- maximum horizontal range from the axis of the rotational unit: 640 mm
- maximum rotation angle of the rotational unit: 330°
- maximum rotation angle of the lower arm: 165°
- maximum rotation angle of the upper arm: 270°
- maximum rotation angle of the wrist: 300°
- maximum lateral rotation angle of the wrist: 270°
- maximum operational speed of the rotational unit, lower arm: 130°/s
- upper arm: 130°/s
- wrist pitch: 130°/s
- wrist rotation: 300°/s
- positioning: incremental counters



Sl. 1. *Dimenzije in območja vrtenja elementov kinematike šolskega robota SLR 1500*
Fig.1. *Dimensions and ranges of the SLR 1500 training robot kinematic elements rotation*

Sl.2. *Šolski robot SPL 1500*
Fig. 2. *The SPL 1500 training robot*

Krmilni sistem robota obsega računalnik z vmesnikom RS 232C. Za računalnik je bila potrebna posebna programska oprema za upravljanje krmilnega sistema. Zahtevam krmilnega sistema je ustrezal programski jezik RAMAS [2]. Ta programski jezik je bil razvit v VUKOV Prešov družbi (Slovaška) za programiranje APR-20

The control system of the robot consists of a computer with an RS 232C interface. For this computer, special software was required for the functions of the control system. The needs of the control system were satisfied with the RAMAS programming language [2]. This programming language was developed in the VUKOV Prešov company (Slovakia)

industrijskega robota s krmilnim sistemom RS-4A. Njegovi ukazi se lahko delijo v več skupin: urejevalnik ukazov, navodila in ukazi za gibanja, tehnološki, krmilni, aritmetični, logični, čakalni ukazi, nato ukazi, ki so namenjeni za ustavljanje, pomožni ukazi in drugi.

1 SIMULIRANJE

Izraz simuliranje običajno pomeni opazovanje in študij obnašanja sistema in temelji na modelu sistema. Sistem sestoji iz urejenega para množic, pri katerem ena od množic pomeni predmete (točke, komponente itn.), druga pa opisuje razmerja med njimi.

Grafični model, ki bi predstavljal obnašanje resničnega robota zelo natančno, bi bil zelo zahteven in neučinkovit. Za simuliranje ne potrebujemo informacij o barvi, hrapavosti ali materialu. Vse bolj pa zahtevamo podatke o izmerah, obliki in značilnosti kinematičnih elementov in seveda o kinematičnih značilnostih razmerja med njimi. Kar je pomembno za simuliranje robota, je torej opis lege elementov glede na dejavnosti robota, za katere je namenjen. Najsi bo to pobiranje predmeta, premikanje zadnjega elementa z ene točke k drugi (z nedoločeno potjo ali z linearno interpolacijo) ali simuliranje njegovega ročnega krmilja.

Za simuliranje robota je bila potrebna informacija o izmerah in značilnostih njegovih kinematičnih elementov. Po poenostavitvi zapletenega simulirnega problema smo opravili naslednje poddejavnosti: določitev kinematične sestave končnih točk v prostoru, tridimenzionalno predstavitev posameznih komponent robota na računalniškem zaslonu, rešitev interpolacije za prostorske premike robota in problem urejanja programa v posebnem jeziku. Za predstavitev komponent robota so bila uporabljena mnogokotna telesa.

Gibanje se izvede po vstopu v programski ukaz s parametri koordinat premika končne točke. Iz teh koordinat so najprej izračunane koordinate robota (rotacije posameznih členkov) in nato se vsi elementi začnejo premikati v ustrezno smer z največjo hitrostjo.

Če je tirnica določena, potem jo tvori interpolator. Interpolacija nadomesti dano tirnico s sistemom koračnih premikov v smeri koordinatnih osi tako, da se izračunana tirnica najbolje ujema z izvirno zahtevano tirnico. Ta tirnica mora zadostiti zahtevam po natančnosti. Še več, od interpolacijske metode pričakujemo zanesljivost, natančnost, preprostost in hitrost interpolacije.

Poznamo dva standardna interpolacijska postopka: digitalno diferencialno interpolacijo (DDI - DDA), ki temelji na diferencialnih enačbah krivulje,

for programming the APR-20 industrial robot with the RS-4A control system. Its commands can be divided into several groups: editing commands, move instructions and commands, technological, control, arithmetical, logical, waiting, interruption serving, help commands and others.

1 SIMULATION

By simulation, we generally understand the observation and study of a system's behaviour based on its model. The system consist of an ordered pair of sets, where one of the sets represents the objects (items, components and so on) and the other describes the relations between them.

A graphical model, which would represent the behaviour of a real robot in great detail would be very difficult and also ineffective. For a simulation we do not need information about the colour, roughness, or material. Rather we require information about the dimension and shape and the character of the kinematic elements and of course about the kinematic character of the relationship between them. What is important for a robot simulation is therefore the depiction its element position with respect to activities the robot is performing. Either the picking up of an object, moving the end element from one point to another (with no trajectory specification or with linear interpolation) or the simulation of its manual control.

For the simulation of the robot information about the dimensions and the characteristics of its kimenatic elements was required. After the simplification of the complex simulation problem, the following main subtasks were encountered: determination of the kinematic structure end points in space, 3D representation of the individual robot components on the computer screen, the interpolation solution for the robot's 3D movement, and the problem of program editing in a special language. Polygonal bodies were used for the representation of the robot components.

The movement is executed after entering a program instruction with the coordinate parameters of the movement end point. From these coordinates the robot coordinates are first calculated (rotations of individual joints) and then all the elements start to move with the maximum speed in the appropriate direction.

If the trajectory is specified, then it is generated by interpolator. The interpolation replaces a given trajectory with a system of incremental movements in the direction of the coordinate axes in such a way that the resulting trajectory is in maximum accordance with the originally requested trajectory. This trajectory must satisfy the requirements for precision. Moreover, from the interpolation method we expect the reliability, accuracy, simplicity and speed of interpolation.

We recognise two standard interpolation procedures: digital differential interpolation (DDA) which is based on the differential equation of the

ki pomeni tirnico, in neposredno računanje funkcije (NRF - DFC), ki temelji na analitičnih izrazih krivulje.

Za simuliranje rokovanja s predmetom mora biti omogočeno odpiranje in zapiranje prijemala. Za predstavitev te operacije uporabljamo vrednost notranjega parametra, ki je namenjen za odpiranje / zapiranje prijemala in hkratno sprostitev / prijemanje predmeta. Stavčni prepis v jezik RAMAS bo torej: FLAG + F0, da se prijemalo zapre in FLAG – F0, da se odpre.

Simuliranje ročnega krmiljenja daje občutek o robotovem gibanju v delovnem okolju. To je vrtenje robota okrog vodoravne osi, vrtenje posameznih ročic okoli lastnih osi, vrtenje prijemalnega mehanizma okoli stranske in vzdolžne osi in tudi odpiranje in zapiranje prijemala. Dalje, ročno krmiljenje robota dovoljuje, da si predstavljamo delovni doseg robota in njegove mejne lege. Gibanje v posamezno smer se sproži s pritiskom na tipke računalniške tipkovnice.

Po poenostavitvi zapletenega simulirnega problema naletimo na naslednje glavne podnaloge:

- določitev kinematične zgradbe končnih točk v prostoru;
- prostorska predstavitev posameznih komponent robota na računalniškem zaslonu (transformacija prostora v ravnino z ohranitvijo vidnosti);
- rešitev z interpolacijo za robotove prostorske gibe;
- problem urejanja programa v jeziku RAMAS, njegov zapis, shranjevanje, nalaganje in izvajanje simuliranja.

Ustvarjanje algoritma sledi razčlembi problema v manjše in preprostejše podprobleme, za katere je laže najti rešitev. Podproblemi so opisani v diagramu poteka. Domnevati je bilo mogoče, da je glavni simulirni program zmožen izpeljati nalogo. Kasneje so bila ta dela do potankosti izvedena v programskem jeziku PASCAL.

Težava se pojavi pri razumevanju programa, ker krmilni jezik RAMAS in programski ukazi za simulirni program niso združljivi. Program RAMAS je ASCII tekstovna datoteka urejena v poljubnem urejevalniku besedila. Morali smo razviti komponento, ki bo rabila za prilagoditev (prevod) programa RAMAS seznamu ukazov, ki jih bo simulirni program razumel.

Algoritem za ta prevajalnik temelji na diagramu poteka. Na temeljih tega diagrama poteka smo izdelali podprogram v jeziku PASCAL, ki prebere program RAMAS, vrstico za vrstico, odkriva ukaz in sledeč njegovim značilnostim in parametrom, pokliče primeren postopek, ki izvede dejavnost natanko po navodilih.

Kot interpolacijski postopek, ki najbolj ustreza našim potrebam, je bila izbrana metoda DDI. Glede na neposredno računanje iz funkcije je njena

curve which represents the trajectory and DFC (Direct functional calculation), which is based on the analytical expression of the curve.

For the simulation of object manipulation it must be possible to ensure opening and closing of the end effector. For the representation of this operation we use a value of internal flag, which will stand for opening / closing of the gripper and at the same time, release / pick-up of the object. The syntactical transcription in the RAMAS language will be then: FLAG +F0 to close the gripper and FLAG -F0 to open the gripper.

The simulation of manual control gives an idea about the robot's movement in the working area. This is the robot's rotation around the horizontal axis, the individual arms rotation around their axes, the gripper mechanism rotation around the lateral and transverse axes and also the opening and closing of the gripper. Further, the manual control of the robot allows us to imagine the robot's working range and its boundary positions. The movement in an individual direction is realised by pressing keys on the computer keyboard.

After the simplification of the complex simulation problem we encounter the following main sub-tasks:

- determination of the kinematic structure end points in space;
- 3D representation of individual robot components on the computer screen (space transformation to a plane with visibility preservation);
- interpolation solution for robot 3D movement;
- problem of program editing in RAMAS language, its writing, saving, loading and execution of the simulation.

The algorithm creation follows the practice of problem decomposition into smaller and simpler sub-problems, for which the solution is easier to find. The sub-problems are described in the flow diagram. The main simulation program was supposed to be able to execute these tasks. Later, these tasks were in detail worked out in the PASCAL programming language.

Since the controlling language is RAMAS and the language instruction for the simulation program are not compatible, we have a problem of program interpretation. The RAMAS program is an ASCII text file edited in an arbitrary text editor. We have had to develop a component, which would serve for adaptation (translation) of the RAMAS program to a list of instructions understandable by the simulation program.

The algorithm for this translator is based on a flow chart. On the basis of this flow chart we have created a subprogram in the PASCAL language, which reads the RAMAS program, line by line, detects the instruction and following its character and parameters, it calls an appropriate procedure, which executes an action particularly for this instruction.

As an interpolation procedure, which best suits our needs, the DDA method was chosen. Its advantage with respect to direct functional calculation is that the DDA

prednost v tem, da interpolator DDI neposredno izračuna funkcijo s stalnimi koraki tirnice. S tem načinom nastaja gibanje iz sunkov in njihovo pogostnost, ki so poslani posamezni osi. Če imamo dve točki, A(x_A, y_A, z_A) in B(x_B, y_B, z_B), mora interpolator tvoriti tirnico med njima z delitvijo tirnice na n odsekov. Na enak način tudi deli na n odsekov tirnico v posamezni osi. Izmere posameznih korakov po osi i_x, i_y, i_z so določene.

Interpolator doseže končno točko giba v vsako smer po izvedbi ustreznega števila korakov. Večje ko je število sunkov na časovno enoto, manjši je korak (inkrement) in natančnost premika je potem večja. Krmilni sistem mora tudi rešiti obratno transformacijo z interpolacijo kartezičnih koordinat v robotske koordinate z visoko frekvenco.

Manj zahteven primer je, če naj bi končni element dosegel končno točko brez opisa tirnice. Za začetne in končne točke se izračunajo lokalne koordinate robota A(φ_{1A}, φ_{2A}, φ_{3A}, φ_{4A}, φ_{5A}) in B(φ_{1B}, φ_{2B}, φ_{3B}, φ_{4B}, φ_{5B}). Za vsak členek se določita smer in kot giba. Če vrednost spremenljivke doseže vrednost kota v končni legi, se gibanje v tej točki ustavi.

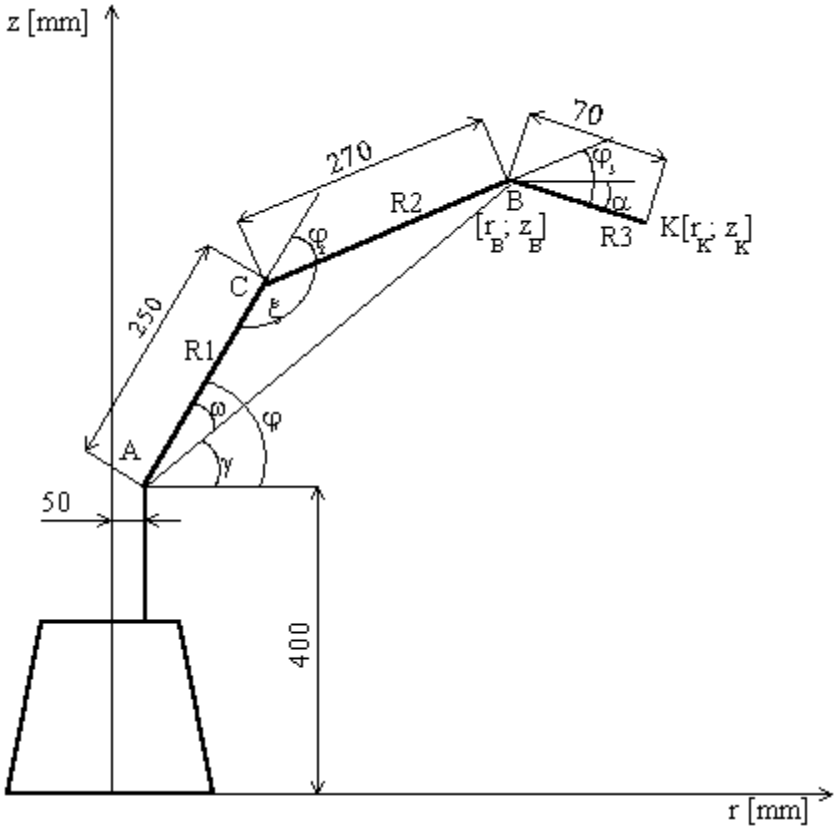
Za programiranje robota potrebujemo razmerje med njegovimi lokalnimi koordinatami in globalnim koordinatnim sistemom in nasprotno in iz znanih globalnih koordinat moramo določiti lego

interpolator directly calculates the function with constant trajectory increments. With this approach, the movement is derived from impulses sent to individual axes related to their frequency. If we have two points, A(x_A, y_A, z_A) and B(x_B, y_B, z_B), the interpolator must generate the trajectory between them by dividing the trajectory into n segments. In the same way, it also divides into n segments a trajectory in each individual axis. The dimensions of individual axes increments i_X, i_Y, i_Z are determined.

After the execution of the appropriate number of steps in each direction the interpolator reaches the end point of the movement. The higher the number of impulses per time unit, the smaller is the unit step (increment) and the movement precision is then also higher. The control system must also solve the inverse transformation from the interpolation Cartesian coordinates to robot coordinates with higher frequency.

A less complicated case is when the robot end element is supposed to reach the end point without trajectory specification. For starting and ending points the robot local co-ordinates are calculated as A(φ_{1A}, φ_{2A}, φ_{3A}, φ_{4A}, φ_{5A}) and B(φ_{1B}, φ_{2B}, φ_{3B}, φ_{4B}, φ_{5B}). For each joint, the direction and the angle of the movement are determined. If the value of a variable reaches the value of the angle in the end position, the movement at this point is terminated.

For the robot programming we need to know the relationship between its local coordinates and the



Sl. 3 *Kinematična shema robota*
Fig. 3. *The robot kinematic scheme*

kinematičnih elementov – lokalnih koordinat robota. Prvi primer se imenuje neposredna transformacija in drugi obratna transformacija.

Za neposredno transformacijo potrebujemo razmerja med koti členkov in kartezičnimi koordinatami, da bi izrazili lego končnega elementa v koordinatnem sistemu, ki je pripet na temelj robota.

Začnemo pri kinematični sestavi robota. Slika 3 prikazuje komponente robota v koordinatnem sistemu, ki ima izhodišče premaknjeno v presečišče med telo in prvo ročico. Izmere posameznih komponent in relativni koti zasukov členkov so pomembni za transformacijo. Vhodne spremenljivke so potem:

– dolžina ročice (določena s konstrukcijo robota) – $R1$, $R2$, $R3$
– relativni koti členkov - φ_1, φ_2, φ_3,
– zasuk robota okoli z osi ψ.

Najprej moramo izračunati absolutne kote med posameznimi ročicami in ravnino *xy*:

$\delta = \varphi_1; \beta = \varphi_1 - \varphi_2; \alpha = \varphi_1 - \varphi_2 - \varphi_3 = \beta - \varphi_3$

in potem transformacijo v cilindrične koordinate:

$r = R_1\cos\delta + R_2\cos\beta + R_3\cos\alpha$, $z = R_1\sin\delta + R_2\sin\beta + R_3\sin\alpha$, $u = \psi$

in nato v kartezične koordinate:

$x = (R_1\cos\delta + R_2\cos\beta + R_3\cos\alpha)\cos\psi$, $y = (R_1\cos\delta + R_2\cos\beta + R_3\cos\alpha)\sin\psi$,

$z = R_1\sin\delta + R_2\sin\beta + R_3\sin\alpha$.

Obratna sredstva transformacija robota je bolj zahtevna in rešitev je mogoča samo za nekatere skupine sorazmerno preprostih kinematičnih sestav. Dokazati je mogoče, da so to v glavnem tiste, pri katerih tri osi zadovoljujejo enega od naslednjih pogojev:

– če so translacijske,
– če so rotacijske z enim sečiščem – sferni členek,
– če so rotacijske z vzporednimi osmi.

Te omejitve vplivajo na oblikovanje mehanskih komponent sodobnih manipulatorjev. Za želeno lego in usmerjenost končnega elementa mehanskega sistema v koordinatnem sistemu, ki je povezan z izhodiščem, je treba določiti lokalne koordinate robota.

Načini, s katerimi lahko izpeljemo grafično simuliranje, so lahko različni:

- Najpreprostejša je simbolična projekcija robota v tri ravnine: od spredaj, s strani in od zgoraj.
- Bolj realističen prikaz prostorskega objekta je aksonometrična predstavitev. Kinematične komponente robota lahko nadomestimo z osnovnimi prostorskimi telesi. Ta telesa so najpogosteje prizme ali telesa, določena s končnim številom ravnih površin – poligoni. Če so ta telesa zapletena in je njihova aksonometrična projekcija nejasna, je treba model prikazati stvarno z upoštevanjem vidnosti robov in ploskev.

Za grafično upodobitev prostorskih

global coordinate system, and also vice versa, and from the known global coordinates we need to determine the position of its kinematic elements - local robot coordinates. The first case is called the direct transformation and the second is the inverse transformation.

For the direct transformation we need to use the relations between joint angles and Cartesian coordinates and to express the end element position in a coordinate system fixed to the robot base.

We start from the robot kinematic structure. Fig. 3 shows the robot components in the coordinate system, the origin of which is translated to the intersection between the waist and the first arm. The dimensions of individual components and relative joint rotation angles are important for the transformation. Input variables are then:

– arm length (determined by robot design) - $R1$, $R2$, $R3$,
– relative joint angles - φ_1, φ_2, φ_3,
– robot rotation around z axis ψ.

First we have to calculate the absolute angles between the individual arms and the plane *xy*:

$\delta = \varphi_1; \beta = \varphi_1 - \varphi_2; \alpha = \varphi_1 - \varphi_2 - \varphi_3 = \beta - \varphi_3$

and then a transformation to cylindrical coordinates:

$r = R_1\cos\delta + R_2\cos\beta + R_3\cos\alpha$, $z = R_1\sin\delta + R_2\sin\beta + R_3\sin\alpha$, $u = \psi$

and then to Cartesian coordinates:

$x = (R_1\cos\delta + R_2\cos\beta + R_3\cos\alpha)\cos\psi$, $y = (R_1\cos\delta + R_2\cos\beta + R_3\cos\alpha)\sin\psi$,

$z = R_1\sin\delta + R_2\sin\beta + R_3\sin\alpha$.

The inverse transformation of the robot is more complicated and the solution is possible only for a certain class of relatively simple kinematic structures. It is possible to prove that they are mainly those, in which the three axes satisfy one of the following conditions:

– they are transitional,
– they are rotational with one intersection - a spherical joint,
– they are rotational with parallel axes.

These constraints influence the design of mechanical components of contemporary manipulators. For the required position and orientation of the end element of the mechanical system in the coordinate system connected with the origin, it is necessary to determine the robot's local coordinates.

Ways in which one can realise a graphical simulation can be different:

– The simplest one is a symbolical projection of the robot to three planes from the front, the side and the top.
– A more realistic representation of 3D object is an axonometric representation. Kinematic components of the robot can be replaced by the basic 3D bodies. Such bodies are frequently the prisms or bodies specified by a finite number of plane surfaces - polygons. If these bodies are complex and their axonometric representation is unclear, it is necessary that the model is represented realistically respecting visible and invisible edges and surfaces.

objektov na ravnini (kot računalniški zaslon) uporabljamo različne metode predstave. Večinoma so središčne, vzporedne ali pravokotne projekcije.

Za prikaz komponent robota uporabljamo poligonalna telesa (stranice so prikazane s poligoni). Računalnik mora delovati enako kakor dejanski objekt v dejanskem okolju, z drugimi besedami, njegova komponenta se mora gibati skladno s kinematičnimi zvezami med njimi. Zato moramo poznati transformacije, ki dovoljujejo telesu, da se giblje v katerokoli točko prostora in vrtenje telesa okoli katerekoli koordinatne osi.

Kinematične komponente robota SLR 1500 so telesa splošne oblike, omejena z ravnimi in krivimi ploskvami. Če simuliranje teče v realnem času, moramo dovoliti nekatere poenostavitve:

- ravne ploskve so prikazane kot poligoni s poljubnim številom oglišč,
- krive ploskve (sferične in cilindrične) so prikazane s poljubnim številom ravnih ploskev.

Če želimo podati dovolj informacij o telesu, moramo prepoznati vsako njegovo stranico, robove in oglišča. S takim sistemom lahko ustvarimo sestavo vhodnih informacij o telesu. Zato je sestavljenja pomembno za računalniško grafiko.

Da bi se grafični model gibal enako kakor dejanski robot, moramo določiti kinematične odvisnosti za grafično prikazovanje ročic. Izpolnjeni morajo biti naslednji pogoji:

- Rotacija ročice 1 spremeni relativni kot (koordinate robota) in absolutni kot med ročico 1 in ravnino *xy*. Druge ročice (2, 3) se vrtijo skupaj z ročico 1.
- Samo njuna absolutna kota α in β se z ravnino *xy* spremenita. Njuna relativna kota z ročico 1 se ne spremenita.
- Vrtenje ročice 2 spremeni relativni kot φ_1 v členku in absolutni kot β. Ročica 3 se vrti skupaj z ročico 2. Samo absolutni kot a se spremeni, relativni kot z ročico 2 se ne spremeni.
- Enako velja tudi za ročico 3 (zapestje) z razliko, da nobeno telo več ne rotira, če ni v zapestju nobenega predmeta.

3 PRINCIPI GRAFIČNEGA PRIKAZA MODELA NA RAČUNALNIŠKEM ZASLONU

Ker smo hoteli ohraniti vidnost teles, so prikazana samo oglišča, pri katerih sta usmerjenost in projekcija enaki kakor usmeritev vhodnih podatkov (samo vidne strani). Točke, ki ustvarjajo posamezne ravnine, se najprej preračunajo (glede na točko opazovanja) na ravnino zaslona (3D v 2D). Za prikaz celotnega modela se moramo odločiti o vidnosti in potem narisati 61 poligonov na zaslon.

For graphical depiction of the 3D object in 2D space (such as computer screen) we use several representation methods. Most of the time they are centre, parallel and right angle projection.

For the robot component representation we use polygonal bodies (their sides are represented by polygons). The computer must behave in the same way as the real object does in the real world, in other words its component must move in accordance with the kinematic relation between them. Therefore, we have to know the transformations, which allow the body to move to any point in space and rotation of the body around any coordinate axis.

The kinematic components of the SLR 1500 robot are bodies of a general shape bounded by planar and curved surfaces. If the simulation is running in real time, we must allow several simplifications:

- planar surfaces are represented by polygons with an arbitrary number of vertices,
- curved surfaces (spherical and cylindrical) are represented by an arbitrary number of planar surfaces.

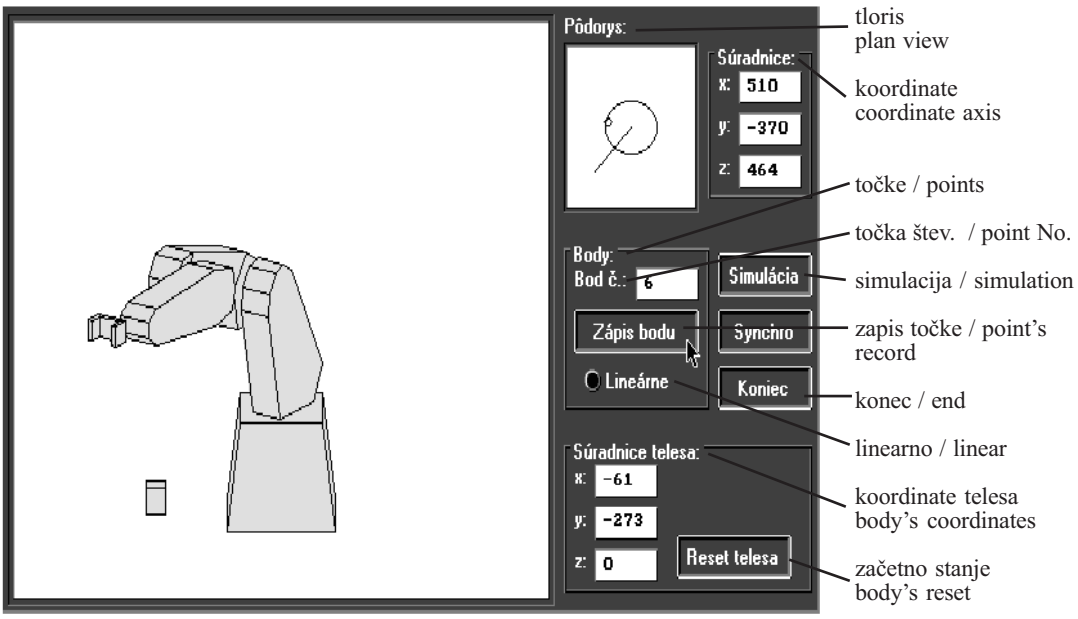
If we want to present sufficient information about the body, we need to encode each of its sides, edges and vertices. With the help of such a system we can create the structure of input information about the body. Therefore, the structuring is important for computer graphics.

In order to keep the graphical model moving in the same way as the real robot, we need to determine the kinematic relationships of graphical objects representing the arms. The following conditions must be satisfied:

- The rotation of the arm 1 changes the relative angle (robot coordinate) and absolute angle, between the arm 1 with the plane *xy*. The remaining arms (2, 3) rotate together with the arm 1.
- Only their absolute angles α and β with the plane xy are changed. Their relative angles with arm 1 do not change.
- The rotation of the arm 2 changes the relative angle φ_1 in the joint and the absolute angle β. The arm 3 rotates together with the arm 2. Only the absolute angle α is changed and the relative angle with arm 2 does not change.
- The same holds also for the arm 3 (wrist), with a difference when no object is picked up, no body rotates any more.

3 PRINCIPLES OF GRAPHICAL MODEL REPRESENTATION ON THE COMPUTER SCREEN

Because we wanted to keep the visibility of the bodies, only the vertices in which the orientation in the projection is same as the orientation of input data are shown (only visible sides). The points which create individual planes are first recalculated (according to the given viewpoint) to the plane of the screen (3D to 2D). For the depiction of the whole



Sl. 4. *Simuliranje šolskega robota SLR 1500*
Fig.4. *The simulation of SLR 1500 training robot*

Hitrost risanja je odvisna od hitrosti računalnika in velikosti in hitrosti njegovega slikovnega pomnilnika.

4 SKLEP

Simulirni program omogoča tvorbo programov za šolski robot SLR 1500 ali z izvajanjem programskih ukazov v jeziku RAMAS ali ročno z učenjem gibanja robota. Tak program je mogoče kasneje urediti z urejevalnikom programov.

Odkar deluje simulirni program s programskim jezikom RAMAS, ki ga uporablja večina robotov slovaške proizvodnje, ga je mogoče uporabiti za poučevanje programiranja industrijskih robotov. Simuliranje se izvede v aksonometrični projekciji in upošteva vidnost robov in ploskev, kar pomaga pri prikazu konkretnega stanja.

Po tem, ko je razvoj krmilnika za robota SLR 1500 končan, je mogoče razširiti te simulirne programe s komunikacijskim modulom. To omogoča pošiljanje razvitih in s simuliranjem preskušenih programov naravnost v pomnilnik krmilnega sistema s serijskim vmesnikom.

model we have to decide about the visibility and then draw 61 polygons to the screen. The speed of drawing depends on the speed of the computer and size and speed of its video memory.

4 CONCLUSION

The simulation program allows us to create the programs for the SLR 1500 training robot either by executing RAMAS language commands or by manually teaching the robot movement. Such a program can be later edited in the program editor.

Since this simulation program works with the RAMAS programming language, which is used in the majority of robots of Slovak production, it is possible to use it for the teaching of industrial robots programming. The simulation is realised in an axonometric view and respects the visibility of the edges and surfaces, which helps us to visualise the concrete situations.

After the development of the hardware control system for the SLR 1500 robot is finished, it is possible to extend this simulation software with a communication module. This would allow as to send the created and simulation verified programs directly to the memory of the control system via a serial interface.

5 LITERATURA
5 REFERENCES

[1] Poppeová , V., Uríček, J., M. Máca (1998) Simulation program of training robot SLR 1500. *Proceedings of the 9th Internationalle DAAAM Symposium.* Cluj-Napoca, Romania.
[2] Programming language RAMAS. (1985) *VUKOV Prešov*, Slovakia.
[3] Mañas, S., Sýkora, P., J. Behalová (1994) Automatizované konstruování II. *ČVUT Praha.*
[4] Kolíbal, Z. (1992) Prumyslové roboty I. *VUT Brno*.
[5] Demeč, P., K. Madáč (1990) Automatizácia inžinierskych prác pri návrhu komponentov progresívnej výrobnej techniky. *Zborník vedeckých prác VŠT Košice,* Alfa Bratislava.

Naslov avtorjev: Dr. Juraj Uríček,
Dr. Viera Poppeová
Róbert Zahoranský
Oddelek za meritve in avtomatizacijo
Fakulteta za strojništvo
Univerze v Žilini
010 26 Žilina, Slovak Republic

Authors' Address: Dr. Juraj Uríček,
Dr. Viera Poppeová
Róbert Zahoranský
Department of Measurement and Automation
Faculty of Mechanical Engineering
University of Žilina
010 26 Žilina, Slovak Republic

Prejeto: 12.7.1999
Received:

Sprejeto: 2.6.2000
Accepted: